



Attorney Docket No. P1189

Patents

Certificate of Express Mailing

Express Mail Mailing Label No. EG 529 200 829 US

Date of Deposit: April 23, 1996

I hereby certify that this paper or fee is being deposited with the United States Postal Service "Express Mail Post Office To Addressee" Service under 37 CFR 1.10 on the date indicated above and is addressed to the Commissioner of Patents and Trademarks, Washington, D. C.

By: Richard A. Jordan
Richard A. Jordan

The Honorable Commissioner of Patents and Trademarks
U.S. Patent and Trademark Office
Washington, D.C. 20231

Dear Sir:

Please find enclosed a patent application as follows:

Applicant(s): Ann M. Wollrath, James H. Waldo, and Roger Riggs

Title: System and Method for Facilitating Dynamic Loading of "Stub" Information to Enable a Program Operating in One Address Space to Invoke Processing of a Remote Method or Procedure in Another Address Space

28 Pages Specification, including 33 Claims and Abstract

6 Sheets Informal Drawings

___ Declaration and Power of Attorney

___ Assignment of invention to: Sun Microsystems, Inc.

___ A check in the amount of \$*** is attached to cover the filing fee.

Basic Fee									\$780.00
-----------	--	--	--	--	--	--	--	--	----------

Additional Fees:

Total Claims	33	, minus	20	=	13	x	\$22.00	=	\$286.00
--------------	----	---------	----	---	----	---	---------	---	----------

Total Ind. Claims	6	, minus	3	=	3	x	\$78.00	=	\$234.00
-------------------	---	---------	---	---	---	---	---------	---	----------

Total Filing Fee									\$1,300.00
------------------	--	--	--	--	--	--	--	--	------------

If this application is found otherwise to be INCOMPLETE, or if at any time it appears that a TELEPHONE CONFERENCE with counsel would helpfully advance prosecution, please telephone the undersigned in Wellesley, Massachusetts, at (617) 431-1357.

Kindly acknowledge receipt of the foregoing application by returning the self-addressed postcard.

Respectfully submitted,

Richard Jordan
Attorney for Applicant
Richard A. Jordan
Reg. No. 27,807

Richard A. Jordan
9 Standish Road
Wellesley, MA 02181-5317
Telephone (617) 431-1357
Fax (617) 235-8326

April 23, 1996



ATTORNEY'S DOCKET NO. P1189
PATENTS

UNITED STATES PATENT APPLICATION

OF

ANN M. WOLLRATH

JAMES H. WALDO

AND

ROGER RIGGS

FOR

SYSTEM AND METHOD FOR FACILITATING DYNAMIC LOADING OF "STUB" INFORMATION TO
ENABLE A PROGRAM OPERATING IN ONE ADDRESS SPACE TO INVOKE PROCESSING OF A
REMOTE METHOD OR PROCEDURE IN ANOTHER ADDRESS SPACE

Certificate of Express Mailing

Express Mail Mailing Label No. EG 529 200 829 US

Date of Deposit April 23, 1996

I hereby certify that this paper or fee is being deposited with the United States Postal Service "Express Mail Post Office To Addressee" Service under 37 CFR 1.10 on the date indicated above and is addressed to the Commissioner of Patents and Trademarks, Washington, D. C. 20231.

By

Richard A. Jordan

Richard A. Jordan

08636706 042396

INCORPORATION BY REFERENCE

The Java™ Language Specification (Sun Microsystems, Inc., 1993-95), (hereinafter referred to as the "Java language specification") a copy of which is attached hereto as Appendix A, incorporated herein by reference.

The Java Virtual Machine Specification (Sun Microsystems, Inc., 1993-95), (hereinafter referred to as the "Java virtual machine specification") a copy of which is attached hereto as Appendix B, incorporated herein by reference.

Ann Wollrath, et al., "A Distributed Object Model for Java™," an unpublished paper attached hereto as Appendix C, incorporated herein by reference.

U. S. Patent Application Ser. No. 08/636,707, filed on even date herewith in the names of James H. Waldo, Krishna Bharat and Roger Riggs, and entitled " System And Method For Generating Identifiers For Uniquely Identifying Object Types For Objects Used In Processing Of Object-Oriented Programs And The Like" (~~Atty. Docket No. P1091~~) (hereinafter identified as the "Waldo, et al., patent application."), ^(now U.S. Patent No. 5,815,709) incorporated herein by reference.

FIELD OF THE INVENTION

The invention relates generally to the field of digital computer systems, and more particularly to systems and methods for facilitating the invocation by a program being processed by a computer in one address space, of processing of methods and procedures in another address space, which may be implemented either on the same computer or on another computer. The invention particularly provides a system and method for obtaining and dynamically loading "stub" information which facilitates invocation by a program operating in one address space of a remote method or procedure in another address space, and possibly on another computer.

BACKGROUND OF THE INVENTION

1
2 In modern "enterprise" computing, a number of personal computers, workstations, and other
3 devices such as mass storage subsystems, network printers and interfaces to the public telephony
4 system, are typically interconnected in one or more computer networks. The personal computers and
5 workstations are used by individual users to perform processing in connection with data and programs
6 that may be stored in the network mass storage subsystems. In such an arrangement, the personal
7 computers/workstations, operating as clients, typically download the data and programs from the
8 network mass storage subsystems for processing. In addition, the personal computers or
9 workstations will enable processed data to be uploaded to the network mass storage subsystems for
10 storage, to a network printer for printing, to the telephony interface for transmission over the public
11 telephony system, or the like. In such an arrangement, the network mass storage subsystems,
12 network printers and telephony interface operate as servers, since they are available to service
13 requests from all of the clients in the network. By organizing the network in such a manner, the
14 servers are readily available for use by all of the personal computers/workstations in the network.
15 Such a network may be spread over a fairly wide area, with the personal computers/workstations
16 being interconnected by communication links such as electrical wires or optic fibers.

17 In addition to downloading information from servers for processing, a client, while processing
18 a program, can remotely initiate processing by a server computer of particular routines and
19 procedures (generally "procedures"), in connection with certain "parameter" information provided
20 by the client. After the server has processed the procedure, it will provide results of its processing
21 to the client, which the client may thereafter use in its processing operations. Typically in such
22 "remote procedure calls" the program will make use of a local "stub" which, when called, transfers
23 the request to the server which implements the particular procedure, receives the results and provides
24 them to the program. Conventionally, the stub must be compiled with the program, in which case
25 the information needed to call the remote procedure must be determined at compile time, rather than
26 at the time the program is run. Since the stub available to the client's programs is static, it may be at

-3-

1 best the closest that can be determined should be provided for the program when it (the program) is
2 compiled. Accordingly, errors and inefficiencies can develop due to mismatches between the stub
3 that is provided to a program and the requirements of the remote procedure that is called when the
4 program is run.

5 SUMMARY OF THE INVENTION

6 The invention provides a new and improved system and method for facilitating the obtaining
7 and dynamic loading of a stub provided to enable a program operating in one address space to
8 remotely invoke processing of a method or procedure in another address space, so that the stub can
9 be loaded by the program when it is run and needed, rather than being statically determined when the
10 program is compiled. Indeed, the stub that is loaded can be obtained from the resource providing the
11 remote method or procedure, and so it (the stub) can exactly define the invocation requirements of
12 the remote method or procedure. Since the stub can be located and dynamically loaded while the
13 program is being run, rather than being statically determined when the program is compiled, run-time
14 errors and inefficiencies which may result from mis-matches between the stub that is provided and
15 the requirements of the remote method or procedure that is invoked can be minimized.

16 In brief summary, the invention provides a stub retrieval and loading subsystem for use in
17 connection with a remote method invocation system. The stub retrieval and loading subsystem
18 controls the retrieval and loading of a stub for a remote method, into an execution environment, to
19 facilitate invocation of the remote method by a program executing in the execution environment. The
20 stub retrieval subsystem includes a stub retriever for initiating a retrieval of the stub and stub loader
21 for, when the stub is received by the stub retriever, loading the stub into the execution environment,
22 thereby to make the stub available for use in remote invocation of the remote method. In one
23 embodiment, the stub retrieval and loading subsystem effects the retrieval and loading for a program
24 operating in one address space provided by one computer, of stub class instances to effect the remote
25 invocation of methods which are provided by objects operating in another address space, which may

-4-

be provided by the same computer or a different computer. In that same embodiment, the stub retrieval and loading subsystem effects the retrieval and loading of a stub class instance when the remote object is referenced, although in other embodiments retrieval and loading may be effected when the remote method is invoked.

BRIEF DESCRIPTION OF THE DRAWINGS

This invention is pointed out with particularity in the appended claims. The above and further advantages of this invention may be better understood by referring to the following description taken in conjunction with the accompanying drawings, in which:

FIG. 1 is a function block diagram of a computer network including an arrangement constructed in accordance with the invention for facilitating the obtaining, dynamic loading and use of "stub" information to enable a program operating in one address space to invoke processing of a remote method or procedure in another address space;

2A through 3B
 FIGS. ~~2 and 3~~ are flow charts depicting the operations performed by the arrangement depicted in FIG. 1, which is useful in understanding the invention, with *FIGs. 2A through 2C* ~~FIG. 2~~ depicting operations performed in connection with obtaining and dynamic loading of the stub information and *FIGs. 3A and 3B* ~~FIG. 3~~ depicting operations performed in connection with use of the stub information to invoke processing of the remote method or procedure.

DETAILED DESCRIPTION OF AN ILLUSTRATIVE EMBODIMENT

FIG. 1 is a schematic diagram of a computer network 10 including an arrangement for facilitating dynamic loading of "stub" information to enable a program operating in one address space to remotely invoke processing of a method or procedure in another address space. With reference to FIG. 1, computer network 10 includes a plurality of client computers 11(1) through 11(N) (generally identified by reference numeral 11(n)), a plurality of server computers 12(1) through 12(M)

1 (generally identified by reference numeral 12(m)), all of which are interconnected by a network
2 represented by a communication link 14. In addition, the network 10 may include at least one
3 nameserver computer 13, which may also be connected to communication link 14, whose purpose
4 will be described below. As is conventional, at least some of the client computers 11(n) are in the
5 form of personal computers or computer workstations, each of which typically includes a system unit,
6 a video display unit and operator input devices such as a keyboard and mouse (all of which are not
7 separately shown). The server computers 12(m) and nameserver computer 13 also typically include
8 a system unit (also not separately shown), and may also include a video display unit and operator
9 input devices.

10 The client computers 11(n), server computers 12(m) and nameserver computer 13 are all of
11 the conventional stored-program computer architecture. A system unit generally includes processing,
12 memory, mass storage devices such as disk and/or tape storage elements and other elements (not
13 separately shown), including network interface devices 15(n), 16(m) for interfacing the respective
14 computer to the communication link 14. The video display unit permits the computer to display
15 processed data and processing status to the operator, and an operator input device enables the
16 operator to input data and control processing by the computer. The computers 11(n) and 12(m) and
17 13 transfer information, in the form of messages, through their respective network interface devices
18 15(n), 16(m) among each other over the communication link 14.

19 In one embodiment, the network 10 is organized in a "client-server" configuration, in which
20 one or more computers, shown in FIG. 1 as computers 12(m), operate as servers, and the other
21 computers, shown in FIG. 1 as computers 11(n) operate as clients. In one aspect, one or more of the
22 server computers 12(m) may, as "file servers," include large-capacity mass storage devices which can
23 store copies of programs and data which are available for retrieval by the client computers over the
24 communication link 13 for use in their processing operations. From time to time, a client computer
25 11(n) may also store data on the server computer 12, which may be later retrieved by it (the client
26 computer that stored the data) or other client computers for use in their processing operations. In
27 addition, one or more of the server computers 12(m) may, as "compute servers," perform certain

1 processing operations in response to a remote request therefor from a client computer 11(n), and
2 return the results of the processing to the requesting client computer 11(n) for use by them (that is,
3 the requesting client computers 11(n)) in their subsequent processing. In either case, the server
4 computers may be generally similar to the client computers 11(n), including a system unit, video
5 display unit and operator input devices and may be usable by an operator for data processing
6 operations in a manner similar to a client computer. Alternatively, at least some of the server
7 computers may include only processing, memory, mass storage and network interface elements for
8 receiving and processing retrieval, storage or remote processing requests from the client computers,
9 and generating responses thereto. It will be appreciated a client computer 11(n) may also perform
10 operations described herein as being performed by a server computer 12(m), and similarly a server
11 computer 12(m) may also perform operations described herein as being performed by a client
12 computer 11(n).

13 The network represented by communication link 14 may comprise any of a number of types
14 of networks over which client computers 11(n), server computers 12(m) and nameserver computers
15 13 may communicate, including, for example, local area networks (LANs) and wide area networks
16 (WANs) which are typically maintained within individual enterprises, the public telephony system,
17 the Internet, and other networks, which may transfer digital data among the various computers. The
18 network may be implemented using any of a number of communication media, including, for example,
19 wires, optical fibers, radio links, and/or other media for carrying signals representing information
20 among the various computers depicted in FIG. 1. As noted above, each of the computers typically
21 includes a network interface which connects the respective computer to the communications link 14
22 and allows it to transmit and receive information thereover.

23 The invention provides a system for facilitating the obtaining and dynamic loading of "stub"
24 information to enable a program operating in one address space to invoke processing of a remote
25 method or procedure in another address space, which may be located on the same computer as the
26 invoking program or on a different computer. The invention will be described in connection with
27 programs provided in the Java™ programming language, as described in the Java language

1 specification, which are processed in connection with an execution environment which is provided
2 by a Java virtual machine. The Java virtual machine, in turn, is specified in the Java virtual machine
3 specification. As described in the Java language specification, programs in the Java programming
4 language define "classes" and "interfaces." Classes are used to define one or more methods or
5 procedures, each of which may be invoked by reference to an interface. A class may be associated
6 with and extend a "super-class," and in that regard will incorporate all of the interfaces and methods
7 of the super-class, and may also include additional interfaces and/or methods. A class may also have
8 one or more sub-classes (and thus will comprise a super-class of each of its sub-classes), with each
9 sub-class incorporating and possibly extending their respective super-classes.

10 An interface provides a mechanism by which a set of methods may be declared. In that
11 connection, an interface identifies each method that is declared by the interface by, for example, a
12 name, identifies the data type(s) of argument(s) that are to be provided for the method, the data
13 type(s) of return values that are to be returned by the method, and identifiers for exceptions which
14 can be thrown during processing of the method. A class may indicate that it implements a particular
15 interface, and in that connection will include the program code which will be used in processing all
16 of the methods which are declared in the interface. In addition, different classes may indicate that
17 they implement the same interface, and each will have program code which will be used in processing
18 all of the methods which are declared in the interface, but the program code provided in each class
19 to for use in processing the methods may differ from the program code provided in the other classes
20 which is used in processing the same methods; thus, an interface provides a mechanism by which a
21 set of methods can be declared without providing an indication of the procedure which will be used
22 in processing any of the methods. An interface may be declared independently of the particular class
23 which implements the method or methods which can be invoked using the interface. In that regard,
24 a class that invokes the method and a class that actually implements the method will not need to share
25 a common super-class.

26 During processing of a Java program, as described in the Java virtual machine specification,
27 a client computer 11(n) provides an execution environment 20 for interpreting the Java program. The

1 Java virtual machine includes a class loader 21 that, under control of a control module 19, can
2 dynamically link instances of classes, generally identified in FIG. 1 by reference numeral 22, into the
3 running program's execution environment while the program is being executed. In that operation,
4 the control module 19 effectively enables the class loader to retrieve uninstantiated classes, which
5 generally identified by reference numeral 23, instantiate them and link them as class instances 22 into
6 the execution environment's address space at the Java program's run time as the methods which the
7 respective classes 23 implement are called. In addition, the class loader 21 can discard ones of the
8 class instances 22 when they are not needed or to conserve memory. It will be appreciated that, if
9 a class instance 22 has been discarded, it may be reloaded by the class loader 21 at a later point if it
10 is then needed.

11 The invention provides an arrangement which facilitates the remote invocation, by a program
12 executing in an execution environment 20 by a client computer 11(n), of methods implemented by
13 classes on a server computer 12(m). In executing a method, the server computer 12(m) will also
14 provide an execution environment 24 for processing, under control of a control module 28, the Java
15 method. In that operation, the Java virtual machine which provides the execution environment 21
16 includes a class loader 25 (which may be similar to the class loader 21) that, under control of the
17 control module 28, can dynamically link an instance of the class 26, to enable the method to be
18 processed in the execution environment 24, and instances of other classes (also generally represented
19 by reference numeral 26) which may be needed to process the remotely-invoked method. In that
20 operation, the control module 28 effectively enables the class loader 25 to retrieve an uninstantiated
21 class for the method to be invoked, from a plurality of uninstantiated classes which are generally
22 identified by reference numeral 27, instantiate it (that is, the uninstantiated class which provides the
23 method to be invoked) and link it as a class instance 26 into the execution environment. In addition,
24 the class loader 25 can discard the class instances 26 when processing of the method has terminated.
25 It will be appreciated that, if class instances 26 has been discarded, it may be reloaded by the class
26 loader 25 at a later point if it is then needed.

1 The structure of nameserver computer 13, if provided, is generally similar to that of the server
2 computer 12(m), and will not be separately described.

3 To facilitate remote invocation of a method, the control module 19 of the client computer's
4 execution environment 21 makes use of one or more stub class instances generally identified by
5 reference numeral 30 which are provided as part of the execution environment 21 in which the various
6 class instances 22, including the class instance which is invoking the remote method, are being
7 processed. Each stub class instance 30 is an instance of an uninstantiated stub class 31, which the
8 server computer 12(m) may maintain for the various class instances 26 and uninstantiated classes 27
9 which the server computer 12(m) has "exported," that is, which the server computer 12(m) makes
10 available to client computers 11(n) for use in remote invocation of methods provided thereby. An
11 uninstantiated stub class 31 includes declarations for the complete set of interfaces for the particular
12 remote uninstantiated class 27 which implements the remote method to be invoked, and also provides
13 or invokes methods which facilitate accessing of the remote method(s) which are implemented by the
14 remote class. The uninstantiated stub class 31, when it is instantiated and provided to the execution
15 environment 20 of the client computer 11(n) as a stub class instance 30, effectively provides the
16 information which is needed by the control module 19 of the execution environment 20 of the
17 invoking Java program, so that, when a remote method that is implemented by its associated class is
18 invoked by a Java program running in a particular execution environment, the remote method will
19 be processed and the return value(s) provided to the invoking Java program. In one embodiment, the
20 arrangement by which the stub class instance may be provided to the execution environment 20 is
21 similar to that described in the aforementioned Waldo, et al., patent application.

22 In addition, the server computer 12(m) provides a skeleton 32, which identifies the particular
23 classes and methods which have been exported by the server computer 12(m) and information as to
24 how it (that is, the server computer 12(m)) may load the respective classes and initiate processing of
25 the particular methods provided thereby.

26 When a class instance invokes a remote method maintained by a server computer 12(m), it
27 will provide values for various parameters to the stub class instance 30 for the remote method, which

1 values the remote method will use in its processing. If the remote method is implemented on the
2 same computer as the invoking Java program, when the invoking Java program invokes a remote
3 method, the computer may establish an execution environment, similar to the execution environment
4 20, enable the execution environment's class loader to load and instantiate the class which implements
5 the method as a class instance similar to class instances 22, and process the remote method using
6 values of parameters which are provided by the invoking class instance in the remote invocation.
7 After processing of the method has been completed, the execution environment in which the remote
8 method has been processed will provide the results to the stub class instance 30 for the remote
9 method that was invoked, which, in turn, will provide to the particular class instance 22 which
10 invoked the remote method.

11 Similar operations will be performed if client computer 11(n) and server computer 12(m) are
12 implemented on different physical computers. In that case, in response to a remote invocation, the
13 client computer 11(n) that is processing the invoking class instance 22, under control of the control
14 module 19 for the execution environment 10 for the invoking class instance 22, will use the
15 appropriate stub class instance 30 to communicate over the network represented by the
16 communication link 14 with the server computer 12(m) which implements the remote method to
17 enable it (that is, the server computer 12(m)) to establish an execution environment 24 for the class
18 which implements the remote method, and to use the class loader 25 to load an instance of the class
19 as a class instance 26. In addition, the client computer 11(n), also using the appropriate stub class
20 instance 30, will provide any required parameter values to the server computer 12(m) over the
21 network 14. Thereafter, the server computer 12(m) will process the remote method using parameter
22 values so provided, to generate result value(s) which are transferred over the network to the client
23 computer 11(n), in particular to the appropriate stub class instance 30. The client computer 11(n)
24 will, after it receives the result value(s) from the network, provide them to the invoking class instance
25 22 for its processing.

26 In any case, when the control module 19 of the client computer's execution environment 20
27 determines that a reference to the remote object has been received, if it determines that the stub class

-11-

1 instance 30 is not present when it receives the reference, it will attempt to obtain the stub class
2 instance 30 from, for example, the server computer 12(m) which implements the remote method, and
3 enable the stub class instance 30 to be dynamically loaded in the execution environment 20 for the
4 invoking class instance 22. A reference to the remote object may be received, for example, either as
5 a return value of another remote method invocation or as a parameter that is received during another
6 remote method invocation. The stub class instance may be dynamically loaded into the execution
7 environment in a manner similar to that used to load class instances 22 in the execution environment
8 22. The execution environment 20 is provided with a stub class loader 33 which, under control of the
9 control module 19, will attempt to find and load the stub class instances 30 as required by the class
10 instances 22 processed in the execution environment. The location of a particular server computer
11 12(m) that maintains the class that implements a method to be invoked remotely may be included in
12 the call from the invoking class instance or may be made known to the stub class loader 33 through
13 another mechanism (not shown) maintained by the client computer 11(n).

14 However, if the stub class loader 33 is not otherwise notified of which server computer 12(m)
15 maintains the class which implements a method which may be invoked remotely, it may use the
16 nameserver computer 13 to provide that identification. The identification may comprise any identifier
17 which may be used to identify a server computer 12(m) or other resource which is available on the
18 network 14 and to which the server computer 12(m) can respond. Illustrative identifiers include, for
19 example, a network address which identifies the server computer and/or resource, or, if the network
20 14 is or includes the Internet, an identifier to, for example, a World Wide Web resource which may
21 provide the identification or a "uniform resource locator" ("URL") which provides a uniform
22 mechanism for identifying resources that are available over the Internet. The server computer 12(m)
23 which implements the remote method, in response to a request from the client computer 11(n) will
24 provide stub class instance 30 which the client computer 11(n) may load into the execution
25 environment 21 to thereafter enable the remote invocation to be initiated.

26 As noted above, if the stub class loader 33 does not know which server computer 12(m)
27 implements the remote method which may be invoked (and thus does not know which computer is

-12-

1 to provide the stub class code for the remote invocation), it may, under control of the control module
2 19, obtain the identification from the nameserver computer 13. In that operation, the stub class
3 loader 33 may use a previously-provided default stub class which is provided for use in such cases.
4 The default class stub, when used by the invoking Java program, enables the computer that is
5 processing the invoking Java program to communicate with the nameserver computer 13 to obtain
6 information which can be used in invoking the remote method. This operation is essentially the same
7 as the invocation of a remote method to be processed by the nameserver computer 13, with the
8 remote method including a parameter identifying the class and method to be remotely invoked, and
9 enabling the nameserver computer 13 to provide the identification of a server computer 12(m) which
10 can process the method to the requesting client computer 11(n) and other information which may be
11 helpful in communicating with the server computer 12(m) and invoking the particular method. It will
12 be appreciated that the nameserver computer 13 will maintain a table (not separately shown) of
13 "exported" resources, that is, resources, such as classes and methods, that are available to client
14 computers 11(n) connected to the network 14, and information, such as the identifications of the
15 particular server computers 12(m) which provide those resources, which will be useful to the client
16 computers 11(n) in making use of the exported resources.

17 It will be appreciated that the nameserver computer 13 may create and maintain the exported
18 resource table in a number of ways that are known in the art. For example, the nameserver computer
19 13 may periodically broadcast requests for exported resource information over the network 14, to
20 which the various server computers 12(m) which maintain exported resources may respond; in that
21 case, the nameserver computer 13 may establish its exported resource table based on the responses
22 from the server computers 12(m). Alternatively, each of the various server computers 12(m) which
23 maintains an exported resource may periodically broadcast information as to the exported resources
24 which it maintains, and the nameserver computer 13 can update its exported resource table based on
25 the broadcasts from the server computer. In addition, the nameserver computer's exported resource
26 table may be established by a system operator and may be fixed until he or she updates it.

1 In any case, the information provided by the nameserver computer 13 in response to a request
 2 initiated by the default stub would include such information as, for example, the identification of a
 3 computer 12(m) which can provide a class which implements the remote method to be invoked,
 4 particular information which the computer (that is, the computer which implements the remote
 5 method) will require to provide the required stub class code, and the like. After receiving the
 6 information from the nameserver computer 13, the computer 11(n) that is processing the invoking
 7 Java program may, under control of the control module 19, use the information communicate with
 8 the computer (that is, the computer which implements the remote method) to obtain the stub class,
 9 and may thereafter invoke the method as described above.

10 With this background, the operations performed by client computer 11(n), server computer
 11 12(m) and, if necessary, nameserver 13 in connection with obtaining and dynamic loading of a stub
 12 class instance when a reference to a remote method is received will be described in connection with
 13 the flow chart depicted in ^{FIGs. 2A through 2C} ~~FIG. 2~~. In addition, operations performed by the client computer 11(n) and
 14 server computer in connection with remote invocation of a method using the stub class instance will
 15 be described in connection with the flow chart depicted in ^{FIGs. 3A and 3B} ~~FIG. 3~~. With reference initially to FIG.
 16 ^{2A} ~~2~~, the execution environment control module 19 will, when it receives a reference to a remote
 17 method, will initially determine whether an appropriate stub class instance is present in the execution
 18 environment 20 to facilitate invocation of the remote method (step 100). If the control module 19
 19 determines that such a stub class instance 30 for the remote method is present in the execution
 20 environment, it may continue other operations (step 101). However, if the control module 19
 21 determines in step 101 that such a stub class instance is not present in the execution environment 20
 22 for the remote method, the control module 19 will use the stub class loader 33 to attempt to locate
 23 and load a stub class instance 30 for the class to process the remote method. In that case, the control
 24 module 19 will initially determine whether the invocation from the class instance 22 included a
 25 resource locator to identify the server computer 12(m) or other resource which maintains the class
 26 for the method to be invoked, or whether it (that is, the control module 19) or the stub class loader
 27 33 otherwise are provided with such a resource locator (step 102). If the control module 19 makes
 28 a positive determination in that step, it will sequence to step 103 to enable the stub class loader 33

-14-

1 to initiate communications with identified server computer 12(m) to obtain stub class instance for the
2 class and method to be invoked (step 103). When the stub class loader 33 receives the stub class
3 instance 30 from the server computer 12(m), it will load the stub class instance 30 into execution
4 environment 20 for the class instance 21 which initiated the remote method invocation call in step 100
5 (step 104). After the stub class instance 30 for the referenced remote method has been loaded in the
6 execution environment, the method can be invoked as will be described below in connection with
a 7 FIGS. 3A and 3B
n FIG. 3.

8 Returning to step 102, if the control module 19 determines that the invocation from the class
9 instance 22 did not include a resource locator to identify the server computer 12(m) or other resource
10 which maintains the class for the method to be invoked, and further that it (that is, the control module
11 19) or the stub class loader 33 is not otherwise provided with such a resource locator, a "class not
12 found" exception may be indicated, at which point the control module 19 may call an exception
13 handler. The exception handler may perform any of a number of recovery operations, including, for
14 example, merely notifying the control module 19 that the remote method could not be located and
15 allow it to determine subsequent operations.

16 Alternatively, the control module 19 may attempt to obtain a resource locator from the
17 nameserver computer 13 or other resource provided by the network 14 (generally represented in FIG.
18 1 by the nameserver computer 13), using a call to, for example, a default stub class instance 30. The
19 call to the default stub class instance 30 will include an identification of the class and method to be
20 invoked and the name of the nameserver computer 13(m). Using the default stub class instance 30,
21 the control module 19 will enable the computer 11(n) to initiate communications with nameserver
22 computer 13 to obtain an identifier for a server computer 12(m) which maintains the class and method
23 to be invoked (step 110). The communications from the default stub class instance 30 will essentially
24 correspond to a remote method invocation, with the method enabling the nameserver computer to
25 provide the identification for the server computer 12(m), if one exists associated with the class and
26 method to be remotely invoked, or alternatively to provide an indication that no server computer
27 12(m) is identified as being associated with the class and method. During the communications in step

-15-

1 110, the default stub class interface 30 will provide, as a parameter value, the identification of class
2 and method to be invoked.

3 In response to the communications from the default stub class instance 30, the nameserver
4 computer 13 will process the request as a remote method (step 111), with the result information
5 comprising the identification for the server computer 12(m), if one exists that is associated with the
6 class and method to be remotely invoked, or alternatively an indication that no server computer 12(m)
7 is identified as being associated with the class and method. After finishing the method, the
8 nameserver computer 13 will initiate communications with the default stub class instance 30 to
9 provide the result information to the default stub class instance 30 (step 112).

10 After receipt of the result information from the nameserver computer 13, the default stub class
11 instance, under control of the control module 19, will pass result information to the stub class loader
12 33 (step 113). Thereafter, the stub class loader 33 determines whether the result information from
13 the nameserver computer comprises the identification for the server computer 12(m) or an indication
14 that no server computer 12(m) is identified as being associated with the class (step 114). If the stub
15 class loader 33 determines that the result information comprises the identification for the server
16 computer 12(m), it (that is, the stub class loader 33) will return to step 101 to initiate communication
17 with the identified server computer 12(m) to obtain stub class instance for the class and method that
18 may be invoked. On the other hand, if the stub class loader 33 determines in step 114 that the
19 nameserver computer 13 had provided an indication that no server computer 12(m) is identified as
20 being associated with the class and method that may be invoked, the "class not found" exception may
21 be indicated (step 115) and an exception handler called as described above.

22 As noted above, the stub class instance 30 retrieved and loaded as described above in
23 connection with ^{FIGs. 2A through 2C} ~~FIG. 2~~ may be used in remote invocation of the method. Operations performed by
24 the client computer 11(n) in connection with remote invocation of the method will be described in
25 connection with the flow chart in ^{FIGs. 3A and 3B} ~~FIG. 3~~. As depicted in FIG. 3^A, when a class instance 22 invokes
26 a method, the control module 19 may initially verify that a stub class instance 30 is present in the
27 execution environment for remote method to be invoked (step 120). If a positive determination is

1 made in step 120, the stub class instance 30 will be used for the remote invocation, and in the remote
2 invocation will provide parameter values which are to be used in processing the remote method (step
3 121). Thereafter, the stub class instance 30 for the remote method that may be invoked will be used
4 to initiate communications with the server computer 12(m) which maintains the class for the remote
5 method (step 122), in the process, the passing parameter values which are to be used in processing
6 the remote method will be passed. It will be appreciated that, if the server computer 12(m) which
7 is to process the method is the same physical computer as the client computer 12(n) which is invoking
8 the method, the communications can be among execution environments which are being processed
9 within the physical computer. On the other hand, if the server computer 12(m) which is to process
10 the method is a different physical computer from that of the client computer 12(n) which is invoking
11 the method, the communications will be through the client computer's and server computer's
12 respective network interfaces 15(n) and 16(m) and over the network 14.

13 In response to the communications from the stub class instance in step 122, the server
14 computer 12(m), if necessary establishes an execution environment 24 for the class which maintains
15 the method that may be invoked, and the uses the information provided by the skeleton 32 to create
16 a class instance 26 for that class (step 123). Thereafter, the server computer 12(m), under control
17 of the control module 28, will process the method in connection with parameter values that were
18 provided by stub class instance 30 (step 124). After completing processing of the method, the server
19 computer 12(m), also under control of the control module 28, will initiate communications with the
20 client computer's stub class instance 30 to provide result information to the stub class instance (step
21 125). In a manner similar to that described above in connection with step 102, if the server computer
22 12(m) which processed the method is the same physical computer as the client computer 12(n) which
23 invoked the method, the communications can be among execution environments 24 and 20 which are
24 being processed within the physical computer. On the other hand, if the server computer 12(m)
25 which processed the method is a different physical computer from that of the client computer 12(n)
26 which is invoking the method, the communications will be through the server computer's and client
27 computer's respective network interfaces 16(m) and 15(n) and over the network 14. After the stub
28 class instance 30 receives the result information from the server computer, it may provide result

1 information to the class instance 22 which initiated the remote method invocation (step 126), and that
2 class instance 22 can continue processing under control of the control module 19.

3 Returning to step 120, if the control module 19 determines in that step that it does not have
4 a stub class instance 30 that is appropriate for the remote method that may be invoked, it may at that
5 point call an exception handler (step 127) to perform selected error recovery operations.

6 The invention provides a number of advantages. In particular, it provides a new system and
7 method for facilitating dynamic loading of a stub which enables a program that is operating in one
8 execution environment to remotely invoke processing of a method in another execution environment,
9 so that the stub can be loaded by the program when it is run and needed. In systems in which stubs
10 are compiled with the program, and thus are statically determined when the program is compiled,
11 they (the stubs) may implement subsets of the actual set of remote interfaces which are supported by
12 the remote references that is received by the program, which can lead to errors and inefficiencies due
13 to mismatches between the stub that is provided to a program and the requirements of the remote
14 procedure that is called when the program is run. However, since, in the dynamic stub loading system
15 and method, the stub that is loaded can be obtained from the particular resource which provides the
16 remote method, it (the stub) can define the exact set of interfaces to be provided to the invoking
17 program at run time, thereby obviating run-time incompatibilities which may result from mis-matches
18 between the stub that is provided and the requirements of the remote method that is invoked.

19 It will be appreciated that a number of modifications may be made to the arrangement as
20 described above. For example, although the execution environment 20 has been described as
21 obtaining and loading stub class instances to facilitate invocation of remote methods when references
22 to the remote methods are received, it will be appreciated that stub class instances may instead be
23 obtained and loaded when the remote methods are initially invoked. Obtaining and loading of the
24 stub class instance for a remote method when a reference thereto is received will have the advantages
25 that (i) the stub class instance will be present in the execution environment when the remote method
26 is actually invoked, and (ii) if the appropriate stub class instance can not be located, the program or
27 an operator may be notified at an early time. On the other hand, obtaining and loading of the stub

-18-

1 class instance for a remote method when the method is to be invoked may result in a delay of the
2 invocation until the correct stub class instance can be found, if the method is in fact not invoked even
3 if a reference to it is received the stub class instance may not need to be located and loaded.

4 It will be appreciated that a system in accordance with the invention can be constructed in
5 whole or in part from special purpose hardware or a general purpose computer system, or any
6 combination thereof, any portion of which may be controlled by a suitable program. Any program
7 may in whole or in part comprise part of or be stored on the system in a conventional manner, or it
8 may in whole or in part be provided in to the system over a network or other mechanism for
9 transferring information in a conventional manner. In addition, it will be appreciated that the system
10 may be operated and/or otherwise controlled by means of information provided by an operator using
11 operator input elements (not shown) which may be connected directly to the system or which may
12 transfer the information to the system over a network or other mechanism for transferring information
13 in a conventional manner.

14 The foregoing description has been limited to a specific embodiment of this invention. It will
15 be apparent, however, that various variations and modifications may be made to the invention, with
16 the attainment of some or all of the advantages of the invention. It is the object of the appended
17 claims to cover these and such other variations and modifications as come within the true spirit and
18 scope of the invention.

19 What is claimed as new and desired to be secured by Letters Patent of the United States is:

CLAIMS

210-b17
1. For use in connection with a remote method invocation system, a stub retrieval and loading subsystem for controlling the retrieval and loading of a stub for a remote method into an execution environment to facilitate invocation of the remote method by a program executing in said execution environment, the stub retrieval subsystem comprising:

A. a stub retriever for initiating a retrieval of said stub; and

B. a stub loader for, when said stub is received by said stub retriever, loading said stub into said execution environment, thereby to make the stub available for use in remote invocation of said remote method.

2. A stub retrieval and loading subsystem as defined in claim 1 further including a remote method reference detector for detecting whether a remote method reference has been received in said execution environment, the stub retriever initiating retrieval of said stub when the remote method reference detector detects that a remote method reference has been received in said execution environment.

3. A stub retrieval and loading subsystem as defined in claim 1 further including a remote method invocation control for controlling invocation of said remote method, said stub retriever initiating retrieval of said stub when the remote method is invoked.

4. A stub retrieval and loading subsystem as defined in claim 1, the remote method invocation system further including a server for processing said remote method in response to a processing request

-20-

therefor, the server further providing said stub in response to a retrieval request from said stub retriever.

5. A stub retrieval and loading subsystem as defined in claim 4 in which server provides a separate address space for processing said remote method from an address space provided by said execution environment.

6. A stub retrieval and loading subsystem as defined in claim 5 in which the address space provided by said server and the address space provided by said execution environment are provided by separate computers.

7. A stub retrieval and loading subsystem as defined in claim 4, further comprising a remote server identifier for providing a server identification for identifying said server.

8. A stub retrieval and loading subsystem as defined in claim 7 further including a remote method reference detector for detecting whether a remote method reference has been received in said execution environment, the remote method reference including a remote method server identifier, the remote server identifier using the remote method server identifier as the server identification.

9. A stub retrieval and loading subsystem as defined in claim 7 further including a remote method invocation control for providing a remote method invocation identification for controlling invocation of said remote method, the remote method invocation providing a remote method server identifier, the remote server identifier using the remote method server identifier as the server identification.

1 10. A stub retrieval and loading subsystem as defined in claim 7, the remote method invocation
2 system further including a nameserver for providing a said server identification, said remote server
3 identifier initiating communication with said nameserver to obtain the server identification for said
4 remote method.

1 11. For use in connection with a remote method invocation method, a stub retrieval and loading
2 method for facilitating the retrieval and loading of a stub for a remote method into an execution
3 environment to facilitate invocation of the remote method by a program executing in said execution
4 environment, the stub retrieval method comprising the steps of:

- 5 A. a stub retrieval step for initiating a retrieval of said stub; and
6 B. a stub loading step for, when said stub is received, loading said stub into said execution
7 environment, thereby to make the stub available for use in remote invocation of said remote
8 method.

1 *30k*
2 *C4* 12. A stub retrieval and loading method as defined in claim 11 further including a remote method
3 reference detection step for detecting whether a remote method reference has been received in said
4 execution environment, the stub retrieval step including the step of initiating retrieval of said stub
when a remote method reference has been received in said execution environment.

1 13. A stub retrieval and loading method as defined in claim 11 further including a remote method
2 invocation control step for controlling invocation of said remote method, said stub retrieval step
3 including the step of initiating retrieval of said stub when the remote method is invoked.

201-1237
14. A stub retrieval and loading method as defined in claim 11, the remote method invocation system further including a server for processing said remote method in response to a processing request therefor, the server further providing said stub in response to a retrieval request from said stub retriever.

15. A stub retrieval and loading method as defined in claim 14 in which server provides a separate address space for processing said remote method from an address space provided by said execution environment.

16. A stub retrieval and loading method as defined in claim 15 in which the address space provided by said server and the address space provided by said execution environment are provided by separate computers.

17. A stub retrieval and loading method as defined in claim 14, further comprising a remote server identification step for providing a server identification for identifying said server.

18. A stub retrieval and loading method as defined in claim 17 further including a remote method reference detection step for detecting whether a remote method reference has been received in said execution environment, the remote method reference including a remote method server identifier, the remote method server identifier being used during the remote method reference detection step as the server identification.

-23-

1 19. A stub retrieval and loading method as defined in claim 17 further including a remote method
2 invocation control step for providing a remote method invocation identification for controlling
3 invocation of said remote method, the remote method invocation providing a remote method server
4 identifier, the remote method server identifier being used during the remote method reference
5 detection step as the server identification.

1 20. A stub retrieval and loading subsystem as defined in claim 17, the remote method invocation
2 system further including a nameserver for providing a said server identification, said remote server
3 identifier initiating communication with said nameserver to obtain the server identification for said
4 remote method.

2004/07/09
1 21. For use in connection with a remote method invocation system, a stub retrieval and loading
2 computer program product for controlling a computer to, in turn, control the retrieval and loading
3 of a stub for a remote method into an execution environment to facilitate invocation of the remote
4 method by a program executing in said execution environment, the stub retrieval computer program
5 product comprising a computer-readable medium having encoded thereon:

- 6 A. stub retriever code devices to enable said computer to initiate a retrieval of said stub; and
7 B. a stub loader code devices to enable said computer to, when said stub is received, loading said
8 stub into said execution environment, thereby to make the stub available for use in remote
9 invocation of said remote method.

1 22. A stub retrieval and loading computer program product as defined in claim 21 further including
2 remote method reference detector code devices for enabling said computer to detect whether a
3 remote method reference has been received in said execution environment, the stub retriever code

-24-

4 devices enabling said computer to initiate retrieval of said stub when the remote method reference
5 detector code devices enable said computer to detect that a remote method reference has been
6 received in said execution environment.

1 23. A stub retrieval and loading computer program product as defined in claim 21 further including
2 remote method invocation control code devices for enabling said computer to control invocation of
3 said remote method, said stub retriever code devices enabling said computer to initiate retrieval of
4 said stub when the remote method is invoked.

15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211

27. A stub retrieval and loading computer program product as defined in claim 24, further comprising
remote server identifier code devices for enabling said computer to provide a server identification for
identifying said server.

28. A stub retrieval and loading computer program product as defined in claim 27 further including
remote method reference detector code devices for enabling said computer to detect whether a
remote method reference has been received in said execution environment, the remote method
reference including a remote method server identifier, the remote server identifier code devices
enabling said computer to use the remote method server identifier as the server identification.

29. A stub retrieval and loading computer program product as defined in claim 27 further including
remote method invocation control code devices for enabling said computer to provide a remote
method invocation identification for controlling invocation of said remote method, the remote method
invocation providing a remote method server identifier, the remote server identifier code devices
enabling said computer to use the remote method server identifier as the server identification.

30. A stub retrieval and loading computer program product as defined in claim 27, the remote method
invocation system further including a nameserver for providing a said server identification, said
remote server identifier code devices enabling said computer to initiate communication with said
nameserver to obtain the server identification for said remote method.

31. For use in connection with a remote method invocation system, a stub retrieval and loading
subsystem for controlling the retrieval and loading of a stub for a remote method into an execution

-26-

environment to facilitate invocation of the remote method by a program executing in said execution environment, the stub retrieval subsystem comprising:

A. a computer; and

B. a control arrangement for controlling said computer, said control arrangement comprising:

i. a stub retrieval module for controlling said computer to initiate a retrieval of said stub; and

ii. a stub loader module for controlling said computer to, when said stub is received in response to said stub retrieval module, load said stub into said execution environment, thereby to make the stub available for use in remote invocation of said remote method.

32. A control arrangement for use in connection with a computer to control the retrieval and loading of a stub for a remote method into an execution environment to facilitate invocation of the remote method by a program executing in said execution environment, said control arrangement comprising:

i. a stub retrieval module for controlling said computer to initiate a retrieval of said stub; and

ii. a stub loader module for controlling said computer to, when said stub is received in response to said stub retrieval module, load said stub into said execution environment, thereby to make the stub available for use in remote invocation of said remote method.

33. A system for distributing code stored on a computer readable medium and executable by a computer, the code including a plurality of modules each configured to control the computer to facilitate the retrieval and loading of a stub for a remote method into an execution environment to

facilitate invocation of the remote method by a program executing in said execution environment, said system comprising:

- i. a stub retrieval module for controlling said computer to initiate a retrieval of said stub; and
- ii. a stub loader module for controlling said computer to, when said stub is received in response to said stub retrieval module, load said stub into said execution environment, thereby to make the stub available for use in remote invocation of said remote method.

ii. a stub loader module for controlling said computer to, when said stub is received in response to said stub retrieval module, load said stub into said execution environment, thereby to make the stub available for use in remote invocation of said remote method.

~~the stub available for use in remote invocation of said remote method.~~

Variable	Mean	SD	Min	Max
Age	38.5	12.5	25	65
Gender	Male	Female		
Marital status	Married	Single		
Education	High school	College		
Occupation	Manager	Worker		
Income	\$10,000	\$20,000		
Health status	Good	Fair		
Exercise frequency	Weekly	Monthly		
Stress level	Low	High		
Sleep quality	Good	Poor		
Dietary habits	Healthy	Unhealthy		
Alcohol consumption	None	Occasional		
Tobacco use	Non-smoker	Smoker		
Family size	2	3		
Work hours	40	50		
Commuting time	30	45		
Home ownership	Owner	Renter		
Neighborhood safety	Safe	Unsafe		
Access to parks	Yes	No		
Public transportation	Good	Poor		
Crime rate	Low	High		
Property value	\$100,000	\$200,000		
Time to school	15	30		
Time to work	20	40		
Time to grocery	10	20		
Time to doctor	15	30		
Time to gym	10	20		
Time to library	15	30		
Time to shopping	10	20		
Time to restaurant	15	30		
Time to park	10	20		
Time to playground	15	30		
Time to community center	10	20		
Time to church	15	30		
Time to synagogue	10	20		
Time to mosque	15	30		
Time to school bus	10	20		
Time to train	15	30		
Time to bus	10	20		
Time to car	15	30		
Time to bike	10	20		
Time to walk	15	30		
Time to wheelchair	10	20		
Time to stroller	15	30		
Time to baby carriage	10	20		
Time to baby walker	15	30		
Time to baby playpen	10	20		
Time to baby crib	15	30		
Time to baby bed	10	20		
Time to baby changing table	15	30		
Time to baby high chair	10	20		
Time to baby stroller	15	30		
Time to baby carriage	10	20		
Time to baby walker	15	30		
Time to baby playpen	10	20		
Time to baby crib	15	30		
Time to baby bed	10	20		
Time to baby changing table	15	30		
Time to baby high chair	10	20		
Time to baby stroller	15	30		
Time to baby carriage	10	20		
Time to baby walker	15	30		
Time to baby playpen	10	20		
Time to baby crib	15	30		
Time to baby bed	10	20		
Time to baby changing table	15	30		
Time to baby high chair	10	20		
Time to baby stroller	15	30		
Time to baby carriage	10	20		
Time to baby walker	15	30		
Time to baby playpen	10	20		
Time to baby crib	15	30		
Time to baby bed	10	20		
Time to baby changing table	15	30		
Time to baby high chair	10	20		
Time to baby stroller	15	30		
Time to baby carriage	10	20		
Time to baby walker	15	30		
Time to baby playpen	10	20		
Time to baby crib	15	30		
Time to baby bed	10	20		
Time to baby changing table	15	30		
Time to baby high chair	10	20		
Time to baby stroller	15	30		
Time to baby carriage	10	20		
Time to baby walker	15	30		
Time to baby playpen	10	20		
Time to baby crib	15	30		
Time to baby bed	10	20		
Time to baby changing table	15	30		
Time to baby high chair	10	20		
Time to baby stroller	15	30		
Time to baby carriage	10	20		
Time to baby walker	15	30		
Time to baby playpen	10	20		

08/636,706

5

A stub retrieval and loading subsystem is disclosed for use in connection with a remote method invocation system. The stub retrieval and loading subsystem controls the retrieval and loading of a stub for a remote method, into an execution environment, to facilitate invocation of the remote method by a program executing in the execution environment. The stub retrieval subsystem includes a stub retriever for initiating a retrieval of the stub and stub loader for, when the stub is received by the stub retriever, loading the stub into the execution environment, thereby to make the stub available for use in remote invocation of the remote method. In one embodiment, the stub retrieval and loading subsystem effects the retrieval and loading for a program operating in one address space provided by one computer, of stub class instances to effect the remote invocation of methods which are provided by objects operating in another address space, which may be provided by the same computer or a different computer.

1990-1991		1991-1992		1992-1993		1993-1994		1994-1995		1995-1996		1996-1997		1997-1998		1998-1999		1999-2000		2000-2001		2001-2002		2002-2003		2003-2004		2004-2005		2005-2006		2006-2007		2007-2008		2008-2009		2009-2010		2010-2011		2011-2012		2012-2013		2013-2014		2014-2015		2015-2016		2016-2017		2017-2018		2018-2019		2019-2020		2020-2021		2021-2022		2022-2023		2023-2024		2024-2025		2025-2026		2026-2027		2027-2028		2028-2029		2029-2030		2030-2031		2031-2032		2032-2033		2033-2034		2034-2035		2035-2036		2036-2037		2037-2038		2038-2039		2039-2040		2040-2041		2041-2042		2042-2043		2043-2044		2044-2045		2045-2046		2046-2047		2047-2048		2048-2049		2049-2050		2050-2051		2051-2052		2052-2053		2053-2054		2054-2055		2055-2056		2056-2057		2057-2058		2058-2059		2059-2060		2060-2061		2061-2062		2062-2063		2063-2064		2064-2065		2065-2066		2066-2067		2067-2068		2068-2069		2069-2070		2070-2071		2071-2072		2072-2073		2073-2074		2074-2075		2075-2076		2076-2077		2077-2078		2078-2079		2079-2080		2080-2081		2081-2082		2082-2083		2083-2084		2084-2085		2085-2086		2086-2087		2087-2088		2088-2089		2089-2090		2090-2091		2091-2092		2092-2093		2093-2094		2094-2095		2095-2096		2096-2097		2097-2098		2098-2099		2099-2100		2100-2101		2101-2102		2102-2103		2103-2104		2104-2105		2105-2106		2106-2107		2107-2108		2108-2109		2109-2110		2110-2111		2111-2112		2112-2113		2113-2114		2114-2115		2115-2116		2116-2117		2117-2118		2118-2119		2119-2120		2120-2121		2121-2122		2122-2123		2123-2124		2124-2125		2125-2126		2126-2127		2127-2128		2128-2129		2129-2130		2130-2131		2131-2132		2132-2133		2133-2134		2134-2135		2135-2136		2136-2137		2137-2138		2138-2139		2139-2140		2140-2141		2141-2142		2142-2143		2143-2144		2144-2145		2145-2146		2146-2147		2147-2148		2148-2149		2149-2150		2150-2151		2151-2152		2152-2153		2153-2154		2154-2155		2155-2156		2156-2157		2157-2158		2158-2159		2159-2160		2160-2161		2161-2162		2162-2163		2163-2164		2164-2165		2165-2166		2166-2167		2167-2168		2168-2169		2169-2170		2170-2171		2171-2172		2172-2173		2173-2174		2174-2175		2175-2176		2176-2177		2177-2178		2178-2179		2179-2180		2180-2181		2181-2182		2182-2183		2183-2184		2184-2185		2185-2186		2186-2187		2187-2188		2188-2189		2189-2190		2190-2191		2191-2192		2192-2193		2193-2194		2194-2195		2195-2196		2196-2197		2197-2198		2198-2199		2199-2200		2200-2201		2201-2202		2202-2203		2203-2204		2204-2205		2205-2206		2206-2207		2207-2208		2208-2209		2209-2210		2210-2211		2211-2212		2212-2213		2213-2214		2214-2215		2215-2216		2216-2217	
-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--	-----------	--

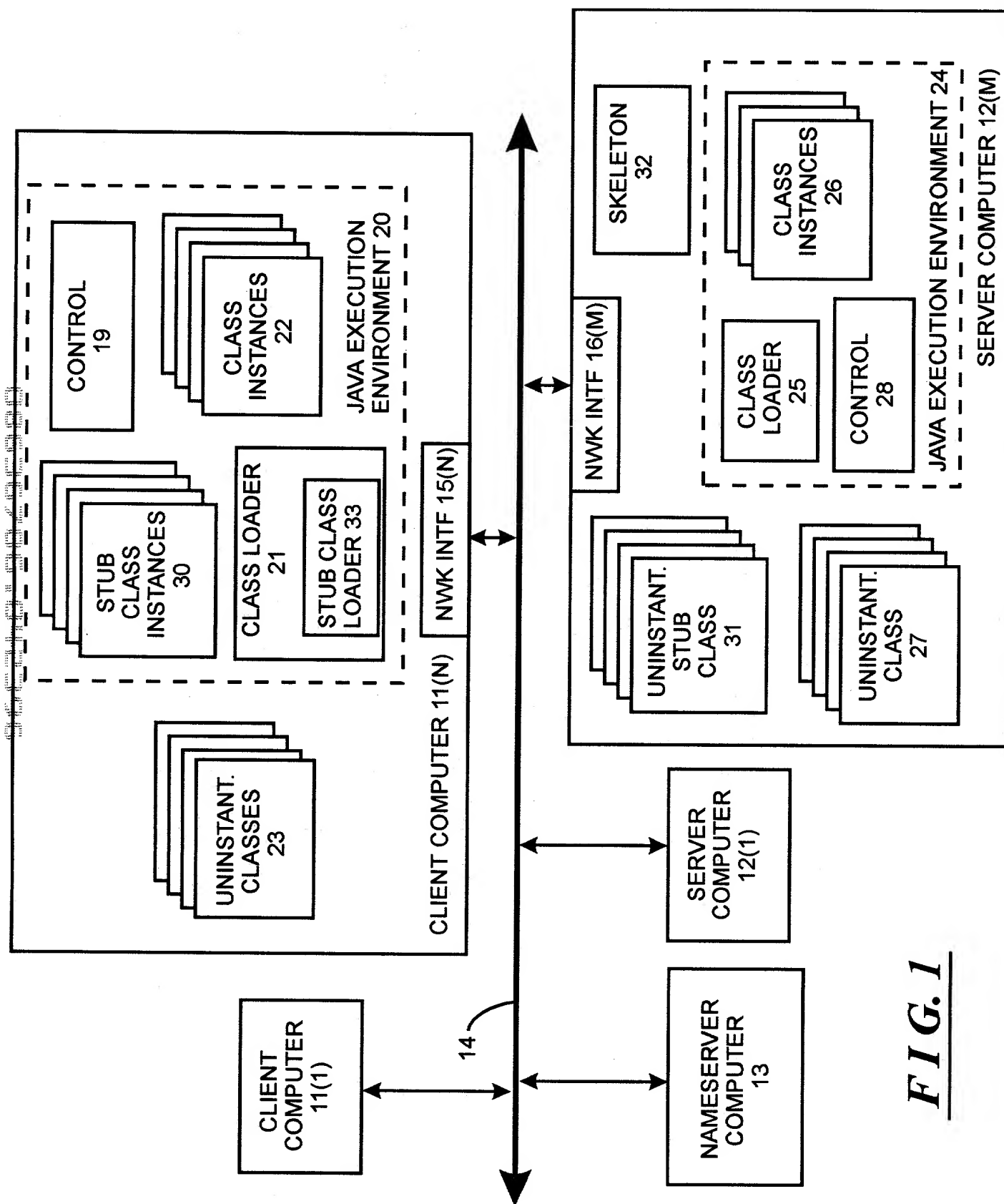
**FIG. 1**

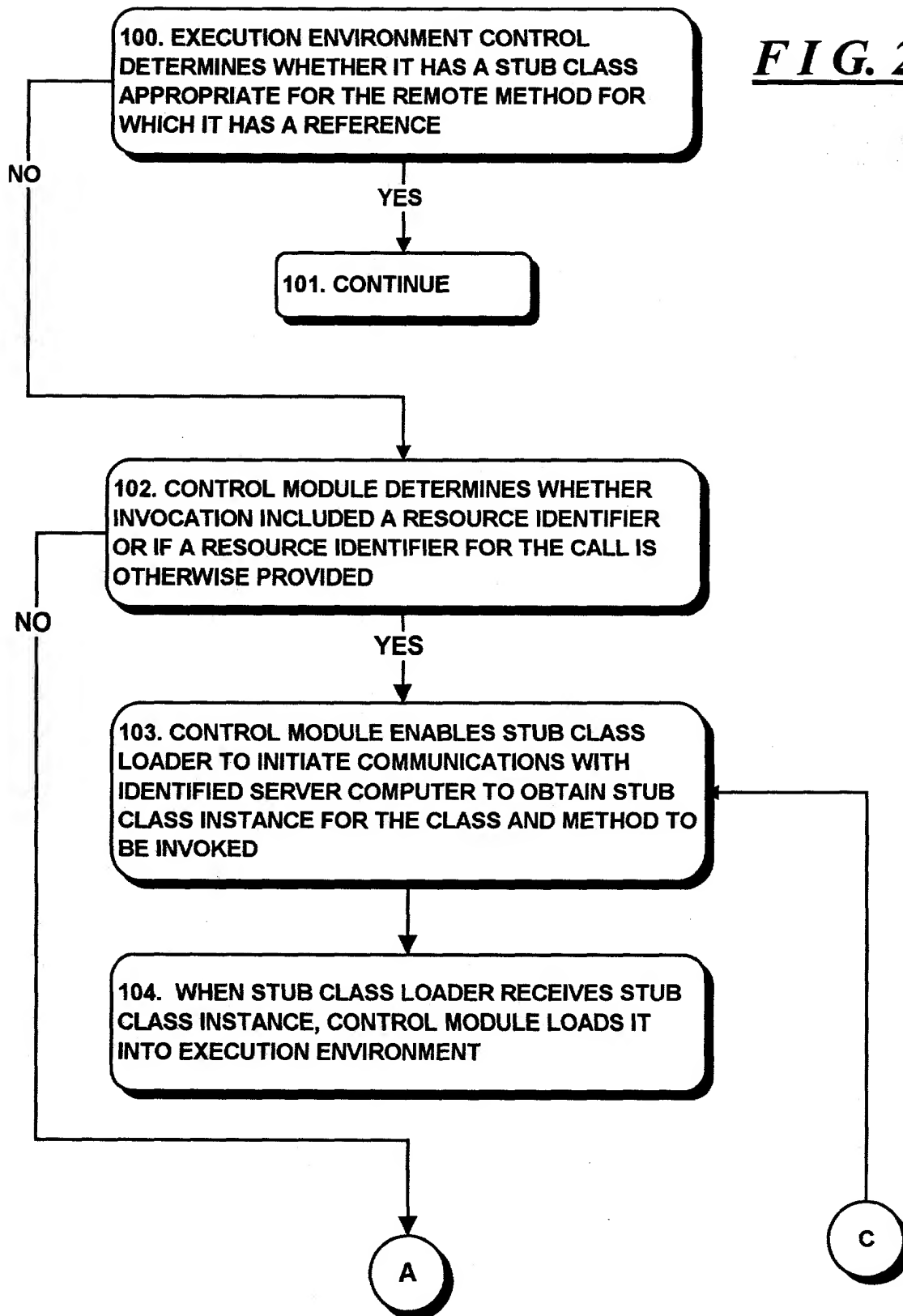
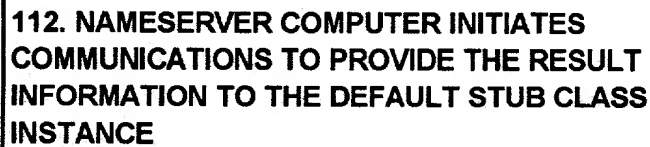
FIG. 2

FIG. 2 (CONT. A)



B

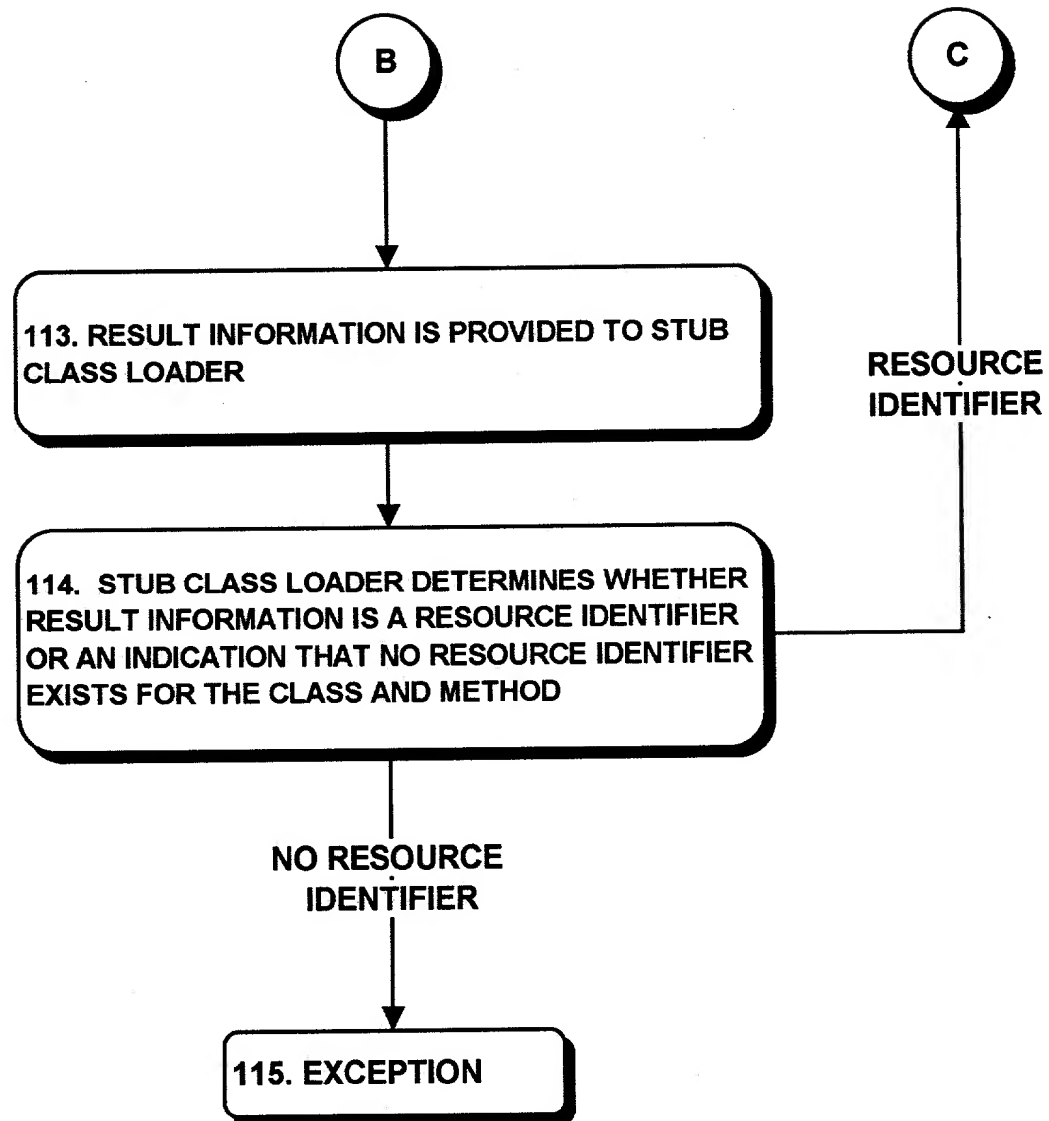
FIG. 2 (CONT. B)

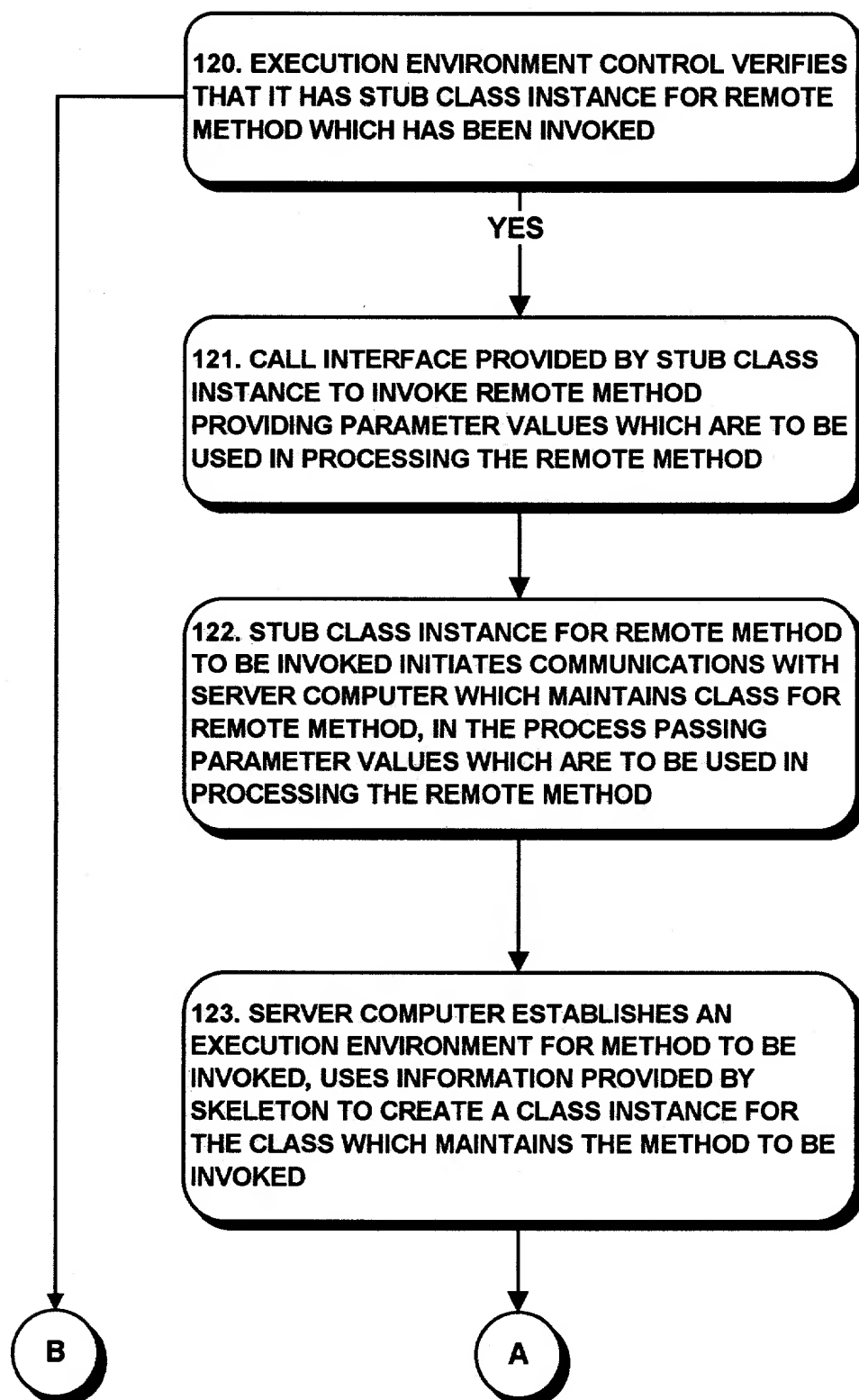
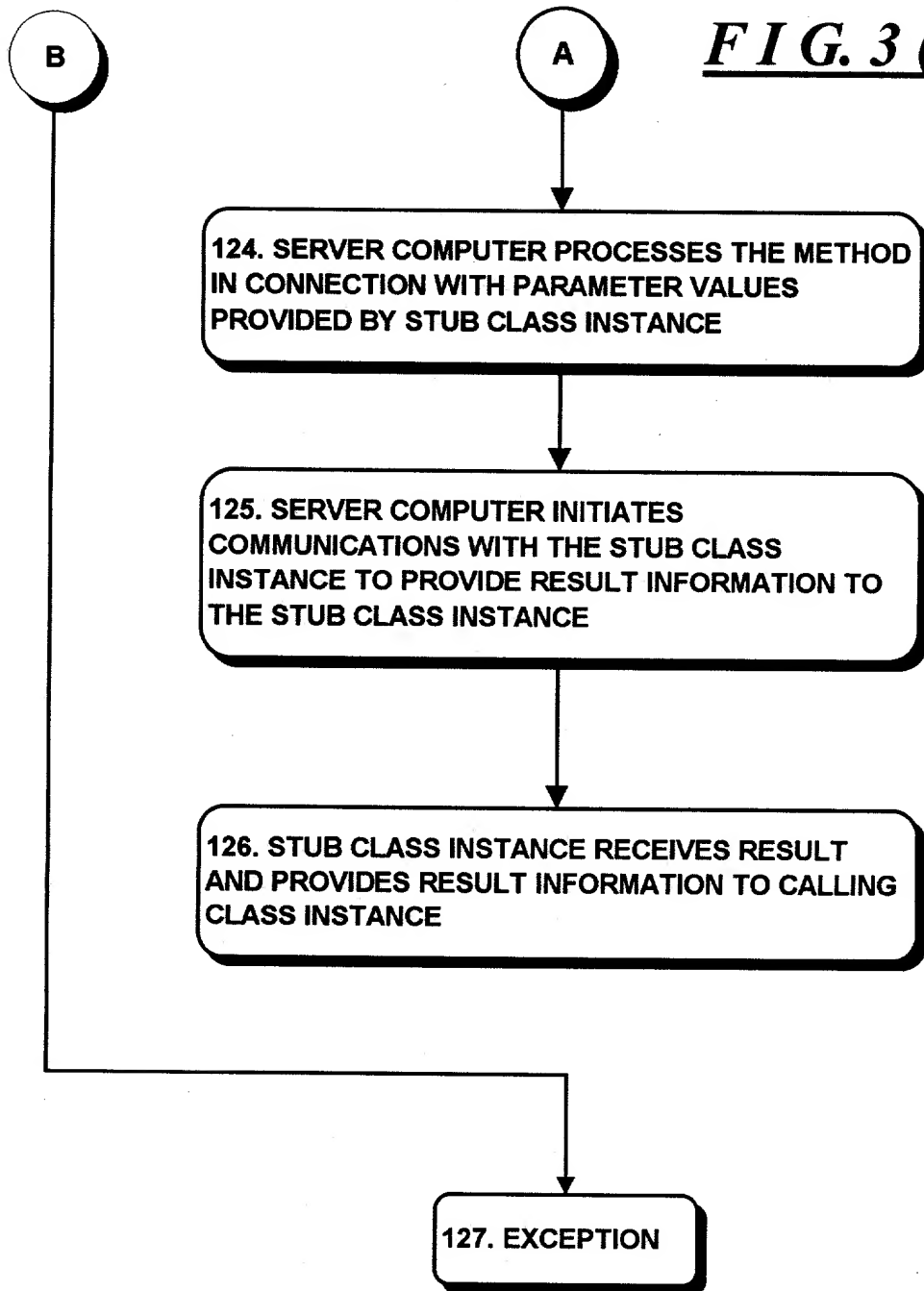
FIG. 3

FIG. 3 (CONT. A)

DECLARATION AND POWER OF ATTORNEY

As a below-named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name.

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled System and Method for Facilitating Dynamic Loading of "Stub" Information to Enable a Program Operating in One Address Space to Invoke Processing of a Remote Method or Procedure in Another Address Space, the specification of which was filed on April 23, 1996, as U. S. Application Serial No. 08/636,706.

I hereby state that I have reviewed and understand the contents of the above-identified application specification, including the claims.

I acknowledge the duty to disclose information that is material to the examination of this application in accordance with Title 37, Code of Federal Regulations, section 1.56(a).

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment or both under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

I hereby appoint:

Richard A. Jordan, Reg. No. 27,807;

Lee Patch, Reg. No. 30,095;

Matthew C. Rainey, Reg. No. 32,291;

James W. Rose, Reg. No. 34,239;

Erwin J. Basinski, Reg. No. 34,773;

Kang S. Lim, Reg. No. 37,491;

Timothy J. Crean, Reg. No. 37,116; and

Leland Z. Wiesner, Reg. No. 39,424,

my attorneys, with full power of substitution, delegation and revocation, to prosecute this application, to make alterations and amendments therein, to receive the patent and to transact all business in the

Patent and Trademark Office connected therewith. Please direct all telephone calls to Richard A. Jordan at (617) 431-1357. Please address all correspondence to Richard A. Jordan, at 9 Standish Road, Wellesley, MA 02181-5317.

Ann M. Wollrath

Ann M. Wollrath

5/16/96
Date

Residence: 9 Northwoods Road
Groton, MA 01450

Citizenship United States of America

Post Office Address: 9 Northwoods Road
Groton, MA 01450

James H. Waldo

James H. Waldo

5/16/96
Date

Residence: 155 Ruby Road
Dracut, MA 01826

Citizenship United States of America

Post Office Address: 155 Ruby Road
Dracut, MA 01826

Roger Riggs

Roger Riggs

5/16/96
Date

Residence: 4 Briarwood Lane
Burlington, MA 01803

Citizenship United States of America

Post Office Address: 4 Briarwood Lane
Burlington, MA 01803

[illegible]